Developing Parallel GeoFEST(P) using the PYRAMID AMR Library

Charles D. Norton, Greg Lyzenga, Jay Parker, and E. Robert Tisdale
Jet Propulsion Laboratory
California Institute of Technology
4800 Oak Grove Drive, Pasadena, CA 91109-8099

***Abstract*— The PYRAMID parallel unstructured adaptive mesh refinement (AMR) library has been coupled with the GeoFEST geophysical finite element simulation tool to support parallel active tectonics simulations. Specifically, we have demonstrated modeling of coseismic and postseismic surface displacement due to a simulated Earthquake for the Landers system of interacting faults in Southern California. The new software demonstrated a 25-times resolution improvement and a 4-times reduction in time to solution over the sequential baseline milestone case. Simulations on workstations using a few tens of thousands of stress displacement finite elements can now be expanded to multiple millions of elements with greater than 98% scaled efficiency on various parallel platforms over many hundreds of processors. Our most recent work has demonstrated that we can dynamically adapt the computational grid as stress grows on a fault. In this paper, we will describe the major issues and challenges associated with coupling these two programs to create GeoFEST(P). Performance and visualization results will also be described.**

I. Introduction

The finite element technique offers nearly the greatest generality in modeling heterogeneous faulted regions of the Earth's crust (e.g., Los Angeles). GeoFEST(P) is an MPI-parallel code which has demonstrated 500 year simulations of postseismic Southern California deformation processes including multiple interacting faults, using 1.4 million finite elements, half-year time steps, and up to 512 processors of various computing systems. Wallclock times are typically a few hours.

A. Overview of GeoFEST

GeoFEST simulates stress evolution, fault slip and plastic/elastic processes in realistic materials [1], [2], [3]. The products of such simulations are synthetic observable time-dependent surface deformation on scales from days to decades. Scientific applications of the code include the modeling of static and transient co– and postseismic Earth deformation, Earth response to glacial, atmospheric and hydrological loading, and other scenarios involving the bulk deformation of geologic media.

It has also been integrated into the QuakeSim portal, which offers users an integrated web services environment. Problems can be specified and solved by non-experts through the web portal, and the resulting deformation can be displayed in combination with Landsat imagery and a digital elevation model.

This research was carried out at the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration.

GeoFEST is designed to aid in interpretation of GPS, InSAR and other geodetic techniques, some of which are undergoing an exponential increase in volume due to NASA remote sensing and satellite geodesy programs. For these, and other reasons, parallelizing GeoFEST represents an important part of the Quakesim Project. For example, simulations are planned that will use a 16 million element model of the Los Angeles basin to find a physical basis for the observed localized compression of the northern portion of the basin. Adaptive mesh refinement capabilities are also needed allowing elements to be concentrated in areas where high resolution is required. To achieve these goals GeoFEST has been coupled with the PYRAMID library.

B. Overview of PYRAMID

Adaptive Mesh Refinement (AMR) is an advanced computational technology applicable to a wide variety of science and engineering simulation problems. The PYRAMID library supports parallel adaptive methods by providing parallel grid-based remeshing techniques. Our software simplifies how the user interacts with the grid for problems that exhibit complex geometry and require the use of parallel computers. PYRAMID contains many library commands, written in Fortran 90, generally organized around the concepts of a mesh data structure and operations that allow manipulation of that structure. All of the concepts about parallel computation are hidden behind library interface commands.

In the following sections we describe the development process associated with coupling GeoFEST and PYRAMID [4], [5] to create GeoFEST(P). Our simulation results modeled the Landers system of faults where the surface nodal mesh is shown in figure 1.

II. Software Development

Working closely with the Quakesim Team we instrumented the GeoFEST software with PYRAMID library commands to support parallelization of the sequential version of the code. This required building a C-adaptor library that allowed GeoFEST (written in C) to use the PYRAMID library (written in Fortran 90). Furthermore, commands from the library for I/O, parallel mesh partitioning, and support for communication of boundary data among processors were added and/or created. During this process optimizations were added to both codes. This software runs on a variety of platforms including both Intel Pentium/Itanium-based and Apple PowerPC-based cluster computers with high efficiency.

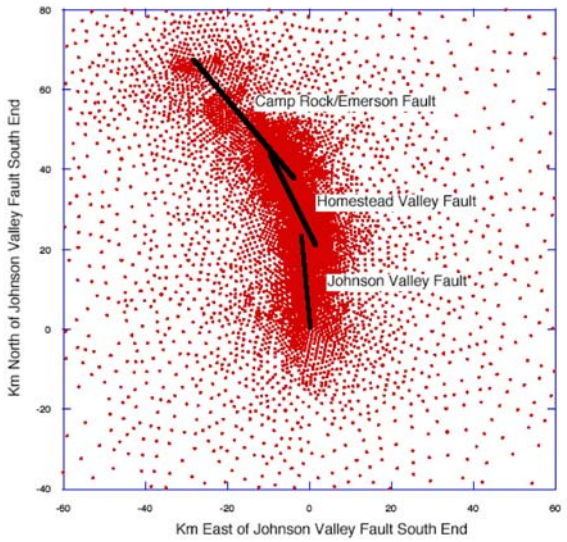

Fig. 1. Vertical fault segments (10 km deep) and surface nodes for LandersGap64 mesh, center region. Node spacing is near 1 km close to the faults, gradually relaxing to 35 km at the boundaries beyond view.

The team used an approach rarely applied in scientific software development called "Extreme Programming". In this method team members collaborate interactively during the design and integration process. This software team development technique consists of one team member editing the software while all members contribute ideas and approaches by viewing the software within a group setting. This method was effective in allowing all team members to bring their specific expertise to the effort that included geophysics, computational science, computer science, and visualization. It also helped minimize errors in the software development as problems were often detected by a team member immediately. Using this method was not an a-priori objective—it simply occurred due to the nature of our collaboration. Four members were involved and this appeared to be an appropriate number in this case.

Nevertheless, numerous challenges faced our team during the development process. These challenges, and their solutions, are presented next.

A. C and Fortran 90 Interoperability

GeoFEST is written in the C programming language while PYRAMID is in Fortran 90 so interoperability was an immediate challenge. The approach we used was *not* based on creating C wrapper software around existing Fortran 90 calls. While this approach is completely portable it would require the use of numerous copies as pointer data structures are passed from Fortran 90 to C.

Instead, a C-adapter library was created that represents an innovation for interlanguage communication between Fortran 90 and C. This is an interface that provides direct access to Fortran 90 data structures from C. This is accomplished by creating an exact replica (in C) of the PYRAMID Fortran 90 mesh data structure and calculating a few compiler specific entities, such as the size, representation, and layout of Fortran 90 pointer types. Although arguments are passed from PYRAMID in Fortran 90 to GeoFEST in C, no copies are ever made and there is effectively no cost for the interaction.

B. Input Data Configuration

Both GeoFEST and PYRAMID require a description of the finite element mesh that is a discretization of the continuous domain. Additionally, the decision was made early on that GeoFEST would use its existing data structures during simulations. This impacts parallel programming development since the GeoFEST input file must be parsed to extract the mesh structure for conversion into PYRAMID format for subsequent parallel mesh partitioning. Additionally, since GeoFEST identifies elements based only on the node positions enrichment of this format is also required as Pyramid requires the specification of faces and edges to support adaptive mesh refinement.

An auxiliary program called "GFmeshparse" was created to translate the GeoFEST input into Pyramid format. (One must be certain that these files are consistent with the problem to be solved.) Since PYRAMID provides the input mesh partitioning this allows GeoFEST to simply store input data associated with that partitioning on a per processor basis. The data structure for GeoFEST is now distributed, but additional work is required to ensure that the finite element solver will perform correctly.

C. Parallel Programming Issues

The parallel version of GeoFEST is designed to be as functionally similar to the original sequential code as possible. From the user perspective, the code is essentially identical, with a few additional steps in order to convert the sequential input file into one that the parallel code can utilize. The basis for the parallel computation performed by GeoFEST is the concept of domain decomposition. The machine model assumed for this style of parallel computing consists of some number of independent processors, each with its own addressable core memory space. The processors are each executing identical code, but not synchronously, as each processor acts and branches in distinct ways on its unique data. The processors interact and exchange data with one another by message passing, and this communication is mediated in the GeoFEST code through use of routines from the PYRAMID library as well as through a small number of direct calls to the MPI protocol.

At the algorithmic level, domain decomposition requires each of the processors to work on a given spatially contiguous piece of the finite element grid. Such communication as is necessary to update and maintain consistency between the subdomains where they join one another is the principal challenge of the parallel programming problem. While the PYRAMID library provides data to GeoFEST describing to the current distribution, routines also provided to handle the inter-partition data communication issues as well. An example of the domain partitioning of the Landers mesh is shown in figure 2.

In the GeoFEST parallel decomposition scheme, each processor has exclusive ownership of a block of finite elements; from this it follows that there will exist shared components such as nodes. These are the nodes that are simultaneously

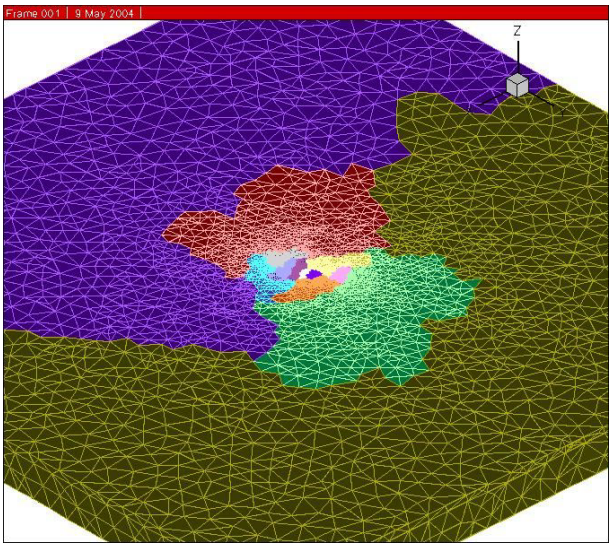

Fig. 2. Landers mesh where colors indicate partitioning among processors (limited to 16 processors in this image for clarity, actually 64 processors were used). Partitions cluster near the domain center due to the high mesh density that is used near the faults.

members of elements that belong to two or more different processors. From this scheme it follows that certain tasks (those which are inherently element-based) can be carried out completely in parallel, without need for interprocessor communication. On the other hand, tasks that are inherently node-based will generally require addition of updating steps that communicate shared nodal information between processors.

The calculation and storage of element stiffness matrix contributions is a task of the first kind; once GeoFEST has been given processor assignments for each element, from PYRAMID's data partitioning, the formation of each element contribution can proceed independently in each processor. However, operations involving the assembled vector of nodal displacements that comprise the fundamental unknowns of the problem are of the second kind. This fact leads us to a decision point in choosing the solver for the global finite element matrix equation which will be an interactive preconditioned conjugate gradient (PCG) solver. This approach has benefits given our domain decomposition strategy and the need to demonstrate scalability for large three-dimensional problems on parallel computers.

The PCG algorithm does not require the stiffness matrix to be assembled in global form; it is sufficient to retain the individual element contributions (and accompanying indexing information) in element-specific storage distributed among processors. As for node-based vectors such as the vectors of displacements and forces, each processor stores that subset of the vectors that correspond to the nodes exclusively within its region, along with redundant storage of all the nodal degrees of freedom that are shared, that is, that are located on a boundary between processor regions.

Note that the two important tasks in the algorithm that require interprocessor communication are the vector dot product and the stiffness matrix-vector product. In the case of the former, each processor calculates its contribution to the global scalar product, using the vector entries in its local storage. This is immediately followed by MPI communication calls that combine the pieces into a global result and distribute the product to all processors. At the conclusion of this operation, each processor is then free to carry on with its independent process.

The matrix-vector product is carried out similarly, although the communication pattern is somewhat more complex. In this task, each processor carries out the multiplication of locally stored matrix elements with locally stored vector entries. The result is usually a vector entry in the local processor, but some of the results will fall on a boundary node which is shared with another processor. In this case, rather than a global (all processors) MPI communication, a pair-wise communication between the involved processors is used to update and reconcile the vector results at all shared nodes, so that at the conclusion of the communication step all processors will contain vector values that agree with one another, and with the values that would be obtained in the equivalent single-processor sequential calculation. The PYRAMID library provides routines to GeoFEST to accomplish these operations.

III. Performance Results and Visualization

The parallel version of the software has given excellent performance results across many different computing systems. The 1992 Landers fault event is modeled consisting of three closely arranged faults with 865 square km in a domain 1000 x 1000 x 60 km as illustrated in figure 2. Figure 3 shows the surface elastic uplift for the Landers fault case for two meshes of different sampling density, for 4 processors (82,000 elements) and for 64 processors (1.4 million elements). The visual quality of the solution is dramatically improved, highlighting the need for meshes with millions of elements and parallel computers.

Figure 5 illustrates the efficiency of the implementation for a scaled problem size on 4, 16, and 64 processors using the Landers data. The performance of the iterative solve (representing the most computationally intensive part of the calculation) across various systems is shown in figure 6. This is measured as the average time per iteration which is a fairly constant metric for scaled problem sizes. A visualization of the coseismic earthquake event followed by the postseismic surface deformation after 500 simulation years is shown in figure 4

Figure 7 shows the computation and communication associated with the iterative solve inner-loop calculation which dominates performance. The black region represents computation of the sparse stiffness matrix/vector product within a local partition (note that the stiffness matrix is not explicitly formed as a single matrix). The red sections represent communication associated with completing that product while the violet represents the global vector dot products associated with data at the shared nodes at inter-partition boundaries. This operation occurs hundreds of times per time-step over hundreds of time steps. The data shows that computation dominates communication and that the calculation is well balanced.

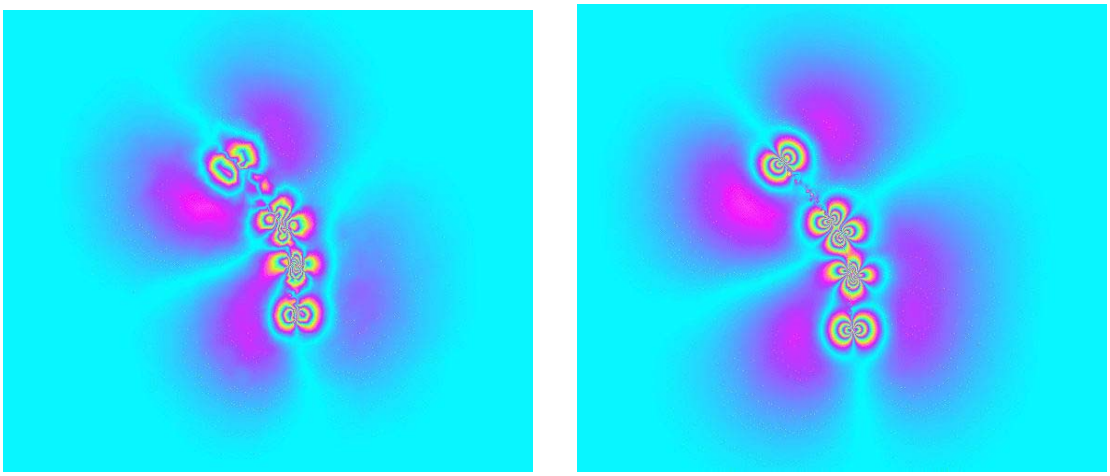

Fig. 3. Models of Landers earthquake deformation at two resolutions. These images show the accuracy improvement going from 82,000 finite elements on four processors (left image) to 1.4 million finite elements on 64 processors (right image).

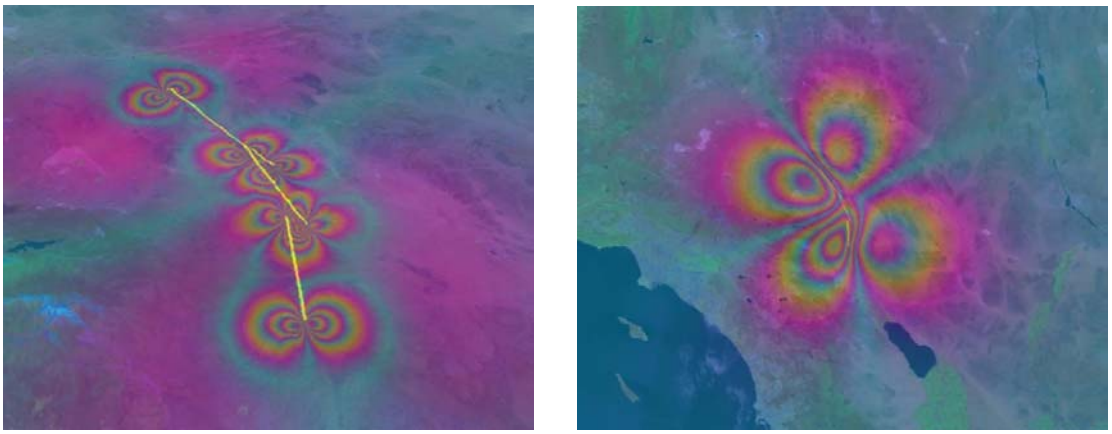

Fig. 4. GeoFEST(P) simulated surface displacement from coseismic Landers model, displayed as InSAR fringes (5.2 cm vertical displacement is one color cycle) with simulated postseismic surface displacement from Landers model after 500 years of viscoelastic relaxation

IV. Future Directions

The team worked closely to instrument the GeoFEST code with PYRAMID library commands. To do this they developed an innovative C-adaptor library to bridge the GeoFEST C code with the Fortran 90 PYRAMID library. Work to date has brought into GeoFEST PYRAMID functions for parallel I/O, parallel mesh partitioning, and support for communication of boundary data among processors. Optimization of key communication functions was added after initial assessment of performance. Porting to a variety of parallel computing platforms ensures the future utility of this combination of GeoFEST and PYRAMID in future developments.

The baseline problem was modeled after the Northridge earthquake (single thrust fault). Analysis of individual earthquake events with attention to their geographic settings is a long-term important area of simulation and research. But with the ability to solve domains with millions of elements implies we can simulate regions with multiple faults, such as the Los Angeles basin. Interactions among slipping faults and possible emergent structures from these nonlinear interactions appears to be the next advance in forecasting earthquake risk. This is a new and promising area for simulation, data comparison and testing of concepts. Many kinds of simulation codes are beginning to be employed for this new kind of work across the earthquake community. But a finite element code has close to the greatest degree of flexibility in including the effects of realistic structures in the Earth in a heterogeneous domain. So we expect this improved GeoFEST will have unique value in validating these other simulations and determining when other multiple-fault models are leaving out too many material effects.

Computationally, we have demonstrated efficient parallel Conjugate Gradient solutions for 3-D faulted-system finite elements, and linked our method with the PYRAMID library. The parallel Conjugate Gradient is no surprise, as it has been demonstrated in other domains of physics. But a freely-available source code for faulted domains will be helpful to the US simulation effort. Linking with PYRAMID is a convenient way to handle issues of partitioning and communication. More important, it is a first step to using the PYRAMID functions for parallel adaptive mesh refinement. Adaptive mesh refinement is essential to attaining high-quality solutions to problems with widely varying stress intensities in three dimensions. Parallel mesh refinement has the additional advantage of solving

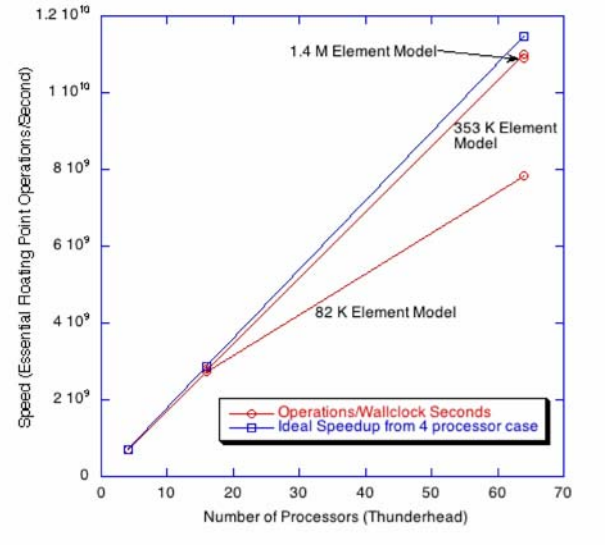

Fig. 5. Scaling of work (for Landers case on linear scales) in GeoFEST(P) time-step function with number of processors on three sizes of problems (on Thunderhead cluster computer, GSFC). Blue indicates ideal scaling (from 4 processors). Expressing work in operations/wallclock time allows comparison of sizes in single plot.

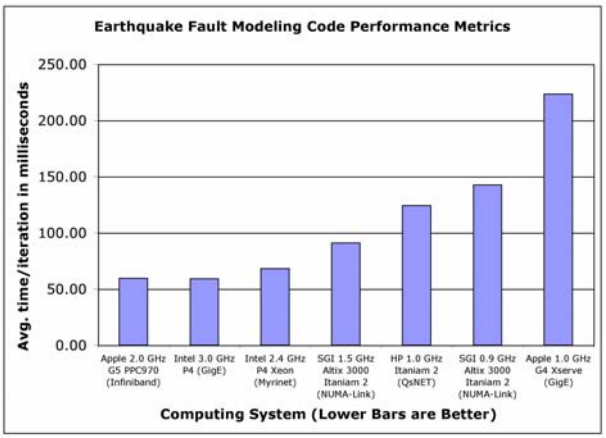

Fig. 6. Performance of iterative solve (most time consuming part of calculation) across various platforms.

problems with size commensurate with the memory space of massively parallel computers without handling the associated mesh files with solid meshing programs, which are nearly all written for sequential machines. A domain can be described and meshed at a relatively low mesh density, imported to the parallel system, and then key locations in the mesh can be refined to the degree needed, expanding the number of elements by large factors (possibly hundreds or more).

In the coming year, the full AMR capability of PYRAMID will be used by QuakeSim. Key features that will be added are implementation and verification of a strain-based indicator of mesh adequacy and assessment of strategies for cycling snapshot solutions from GeoFEST with a limited number of PYRAMID refinements. Using AMR, we expect to use GeoFEST for long-term simulations, with refined meshes that make near-optimal use of over 16 million finite elements.

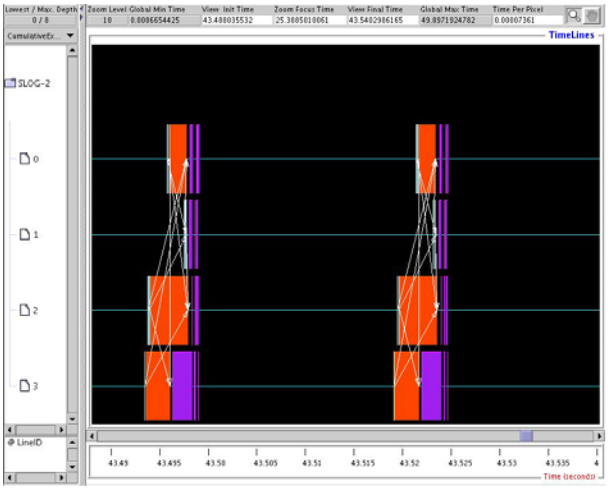

Fig. 7. Fine-detail image of the portion of a GeoFEST run that represents most of the computer time, indicating good parallel performance. Four processors are represented along the vertical axis, time (total 50 ms) along the horizontal axis. Two iterations of the Conjugate Gradient algorithm are shown, of the thousands of iterations making up this simulation. Three features indicate this algorithm will scale to very large problems: 1) The computational load (thin horizontal turquoise line on black background) is about the same for each processor. 2) The time spent in synchronization and communication (red and violet) is a small fraction of the total. The white arrows indicate the inter-processor communication paths among processors where such communication occurs only as needed. (This explains why some processors spend more time in synchronization than others even though the fraction of time is small.) 3) (not visible in this plot) the fraction of time spent in communication does not grow when problem size grows proportional to the number of processors used.

In this way users will be able to model exquisite details of complex regional active tectonics.

Acknowledgments

We acknowledge the on-going support of the ESTO/CT Program and numerous others that have contributed their expertise to the development of this software. This research was carried out at the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration.

References

[1] J.W. Parker, A. Donnellan, G. Lyzenga, J.B. Rundle, and T. Tullis, "Performance Modeling Codes for the QuakeSim Problem Solving Environment, Computational Science," in *Proc. International Conference on Computational Science (Part III)*, Berlin, 2003, pp. 845–862, Springer–Verlag.

[2] A. Donnellan, G. Lyzenga, J. Parker, C. Norton, M. Glasscoe, and T. Baker, *GeoFEST User's Guide*, 2003, http://www.openchannelsoftware.org/.

[3] A. Donnellan et. al., "Numerical Simulations for Active Tectonics Processes: Increasing Interoperability and Performance," Tech. Rep., JPL, http://quakesim.jpl.nasa.gov/, Aug 2003.

[4] C. D. Norton and T. A. Cwik, "Early Experiences with the Myricom 2000 Switch on an SMP Beowulf Class Cluster for Unstructured Adaptive Meshing," in *2001 IEEE International Conference Conference on Cluster Computing*, D. Katz and et. al. T. Sterling, Eds., Newport Beach, CA, October 8-11 2001, IEEE Task Force on Cluster Computing, IEEE Computer Society.

[5] C. D. Norton and T. A. Cwik, "Parallel Unstructured AMR and Gigabit Networking for Beowulf-Class Clusters," *Lecture Notes in Computer Science*, vol. 2328, 2002.